

IRV ENGLANDER • WILSON WONG

The Architecture of Computer Hardware, Systems Software, and Networking

An Information Technology Approach

SIXTH EDITION



WILEY

The Self-Operating Napkin

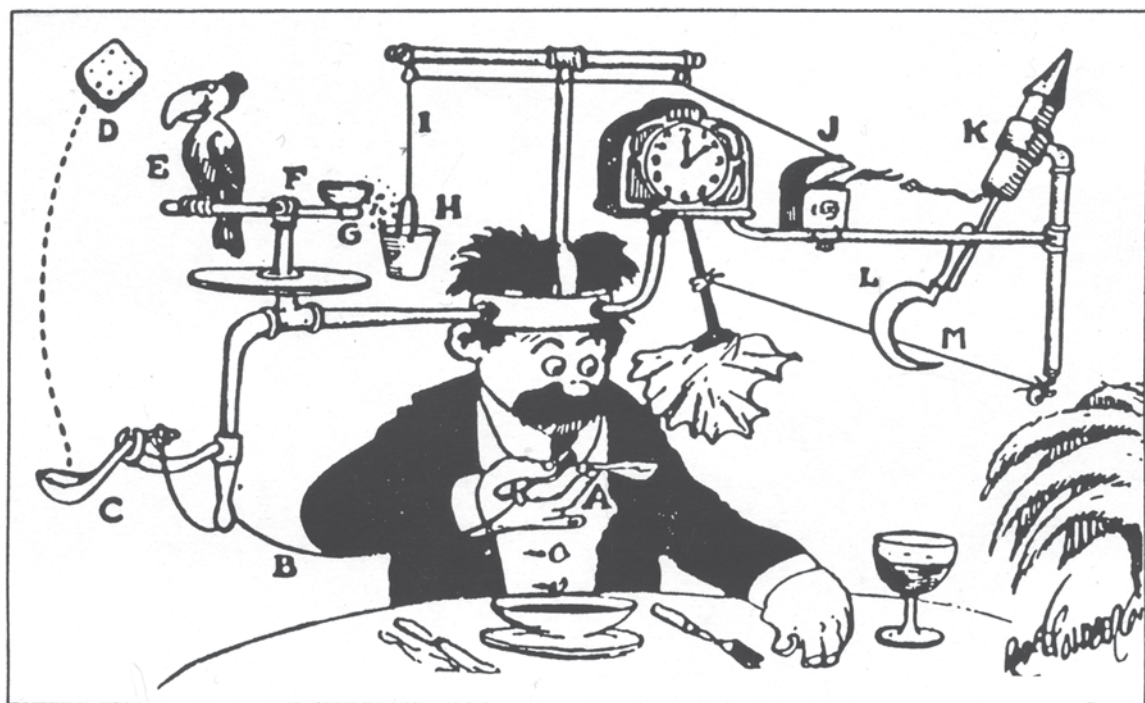
The professor walks in his sleep, strolls through a cactus field in his bare feet, and screams out an idea for self-operating napkin.

As you raise spoon of soup (A) to your mouth it pulls string (B), thereby jerking ladle (C) which throws cracker (D) past parrot (E). Parrot jumps after cracker and perch (F) tilts, upsetting seeds (G) into pail (H). Extra weight in

pail pulls cord (I) which opens and lights automatic cigar lighter (J), setting off sky-rocket (K) which causes sickle (L) to cut string (M) and allow pendulum with attached napkin to swing back and forth thereby wiping off your chin.

After the meal, substitute harmonica for the napkin and you'll be able to entertain the guests with a little music.

Rube Goldberg™ and a © of Rube Goldberg, Inc.
Distributed by United Media.



THE ARCHITECTURE OF COMPUTER HARDWARE, SYSTEMS SOFTWARE, & NETWORKING		

THE ARCHITECTURE OF COMPUTER HARDWARE, SYSTEMS SOFTWARE, & NETWORKING

SIXTH
EDITION

AN INFORMATION TECHNOLOGY
APPROACH

Irv Englander

Bentley University

and

Wilson Wong

Worcester Polytechnic Institute

WILEY

VP AND EDITORIAL DIRECTOR
PUBLISHER
EDITOR
EDITORIAL ASSISTANT
SENIOR MANAGING EDITOR
PRODUCTION EDITOR
ASSISTANT MARKETING MANAGER
COVER PHOTO CREDIT

Mike McDonald
Lise Johnson
Jennifer Manias
Kali Ridley
Judy Howarth
Umamaheswari Gnanamani
Rachel Karach
© Norm Christiansen

This book was set in 10/12 pt STIX Two Text by SPi Global.

Founded in 1807, John Wiley & Sons, Inc. has been a valued source of knowledge and understanding for more than 200 years, helping people around the world meet their needs and fulfill their aspirations. Our company is built on a foundation of principles that include responsibility to the communities we serve and where we live and work. In 2008, we launched a Corporate Citizenship Initiative, a global effort to address the environmental, social, economic, and ethical challenges we face in our business. Among the issues we are addressing are carbon impact, paper specifications and procurement, ethical conduct within our business and among our vendors, and community and charitable support. For more information, please visit our website: www.wiley.com/go/citizenship.

Copyright © 2021, 2014, 2009, 2003 John Wiley & Sons, Inc. All rights reserved. No part of this publication may be reproduced, stored in a retrieval system or transmitted in any form or by any means, electronic, mechanical, photocopying, recording, scanning or otherwise, except as permitted under Sections 107 or 108 of the 1976 United States Copyright Act, without either the prior written permission of the Publisher, or authorization through payment of the appropriate per-copy fee to the Copyright Clearance Center, Inc. 222 Rosewood Drive, Danvers, MA 01923, website www.copyright.com. Requests to the Publisher for permission should be addressed to the Permissions Department, John Wiley & Sons, Inc., 111 River Street, Hoboken, NJ 07030-5774, (201)748-6011, fax (201)748-6008, website <http://www.wiley.com/go/permissions>.

Evaluation copies are provided to qualified academics and professionals for review purposes only, for use in their courses during the next academic year. These copies are licensed and may not be sold or transferred to a third party. Upon completion of the review period, please return the evaluation copy to Wiley. Return instructions and a free of charge return mailing label are available at www.wiley.com/go/returnlabel. If you have chosen to adopt this textbook for use in your course, please accept this book as your complimentary desk copy. Outside of the United States, please contact your local sales representative.

ISBN: 978-1-119-49520-8 (PBK)
ISBN: 978-1-119-49492-8 (EVALC)

Library of Congress Cataloging-in-Publication Data

Names: Englander, Irv (Professor Emeritus),
author. | Wong, Wilson (Professor), author. |
John Wiley & Sons, Inc., publisher.
Title: The architecture of computer hardware, systems software, &
networking : an information technology approach / Irv Englander, Bentley
University ; Wilson Wong, Worcester Polytechnic Institute.
Description: Sixth edition. | Hoboken, NJ : Wiley, [2021] | Includes index.
Identifiers: LCCN 2020044519 (print) | LCCN 2020044520 (ebook) | ISBN
9781119495208 (paperback) | ISBN 9781119494935 (adobe pdf) | ISBN
9781119494843 (epub)
Subjects: LCSH: Computer architecture. | Systems software. | Computer
networks.
Classification: LCC QA76.5 .E566 2021 (print) | LCC QA76.5 (ebook) | DDC
004.2/2—dc23
LC record available at <https://lcn.loc.gov/2020044519>
LC ebook record available at <https://lcn.loc.gov/2020044520>

The inside back cover will contain printing identification and country of origin if omitted from this page. In addition, if the ISBN on the back cover differs from the ISBN on this page, the one on the back cover is correct.

To four outstanding teachers and great human beings:

With your guidance, inspiration, and patience, you showed me that everything is possible.

Dr. Sidney H. Englander (1900–1980) and Mildred K. Englander (1906–2008), in

memoriam my father and mother

Albert L. Daugherty, in memoriam teacher of Science in Cleveland Heights, Ohio from

1927 to 1970

Edith B. Malin, in memoriam teacher of English in Cleveland Heights, Ohio from 1924

to 1958

For the most important people in my life who showed me unwavering support and

the amazing mentors and teachers who I learned from along the way:

Wilson Wong-Ching and Irma Lau de Wong

Brian O’Keeffe

Dr. Irv Englander

Dr. Amy Williams Beers

BRIEF CONTENTS

Preface	xxiii
About the Authors	xxix
PART I AN OVERVIEW OF COMPUTER SYSTEMS	
1 Computers and Systems	4
2 An Introduction to System Concepts and Systems Architecture	36
PART II DATA IN THE COMPUTER	
3 Number Systems	52
4 Data Formats	78
5 Representing Numerical Data	110
PART III COMPUTER ARCHITECTURE AND HARDWARE OPERATION	
6 The Little Man Computer	152
7 The CPU and Memory	166
8 CPU and Memory: Design, Enhancement, and Implementation	200
9 Input/Output and Buses	230
10 Computer Peripherals	262
11 Computer System Organization	298
PART IV NETWORKS, DATA COMMUNICATIONS, AND NETWORKED COMPUTER SYSTEMS	
12 Networks and Data Communications—An Overview	318
13 Ethernet and TCP/IP Networking	356
14 Communication Channel Technology	396
15 Modern Networked Computer Systems	426
PART V THE SOFTWARE COMPONENT	
16 Operating Systems: An Overview	454
17 The User View of Operating Systems	486

18	File Management	518
19	The Internal Operating System	556
	Bibliography	B-1
	Index	I-1
	Supplementary Chapters	
	<i>On the Web at www.wiley.com/Englander6e</i>	
	SUPPLEMENTARY CHAPTER 1	An Introduction To Digital Computer Logic
	SUPPLEMENTARY CHAPTER 2	System Examples
	SUPPLEMENTARY CHAPTER 3	Instruction Addressing Modes
	SUPPLEMENTARY CHAPTER 4	Programming Tools

CONTENTS

PREFACE	xxiii
ABOUT THE AUTHORS	xxix

PART I AN OVERVIEW OF COMPUTER SYSTEMS

1 COMPUTERS AND SYSTEMS 4

1.0	Introduction,	5
1.1	The Starting Point,	8
1.2	Components of the Computer System,	10
	The Hardware Component,	13
	The Software Component,	15
	The Communication Component,	17
	The Computer System,	17
1.3	The Concept of Virtualization,	19
1.4	Protocols and Standards,	20
1.5	Overview of This Book,	21
1.6	A Brief Architectural History of the Computer,	22
	Early Work,	23
	Computer Hardware,	24
	Operating Systems,	27
	Communication, Networks, and the Internet,	31
	Summary and Review,	32
	For Further Reading,	33
	Key Concepts and Terms,	33
	Reading Review Questions,	33
	Exercises,	34

2 AN INTRODUCTION TO SYSTEM CONCEPTS AND SYSTEMS ARCHITECTURE 36

2.0	Introduction,	37
2.1	The General Concept of Systems,	37
2.2	IT System Architectures,	44
	The Role of the System Architect,	46
	Summary and Review,	48
	For Further Reading,	48

Key Concepts and Terms, 49
Reading Review Questions, 49
Exercises, 49

PART II DATA IN THE COMPUTER

3 NUMBER SYSTEMS 52

3.0 Introduction, 53
3.1 Numbers as a Physical Representation, 53
3.2 Counting in Different Bases, 54
3.3 Performing Arithmetic in Different Number Bases, 59
3.4 Numeric Conversion Between Number Bases, 63
 Alternative Conversion Methods, 65
3.5 A Special Conversion Case—Number
 Bases That Are Related, 67
3.6 Fractions, 68
 Fractional Conversion Methods, 71
3.7 Mixed Number Conversions, 73
Summary and Review, 73
For Further Reading, 74
Key Concepts and Terms, 74
Reading Review Questions, 74
Exercises, 75

4 DATA FORMATS 78

4.0 Introduction, 79
4.1 General Considerations, 79
4.2 Alphanumeric Character Data, 82
4.3 Visual Data, 88
 Bitmap Images, 89
 Object Images, 93
 Representing Characters as Images, 96
 Video Images, 96
4.4 Audio Data, 97
4.5 Data Compression, 100
4.6 Page Description Languages, 102
4.7 Internal Computer Data Format, 102
 Numerical Character to Integer Conversion, 104
Summary and Review, 105
For Further Reading, 106
Key Concepts and Terms, 106
Reading Review Questions, 106
Exercises, 107

5 REPRESENTING NUMERICAL DATA 110

- 5.0** Introduction, 111
- 5.1** Unsigned Binary and Binary-Coded Decimal Representations, 111
- 5.2** Representations for Signed Integers, 115
 - Sign-and-Magnitude Representation, 115
 - Nine's Decimal and 1's Binary Complementary Representations, 117
 - Ten's Complement and 2's Complement, 124
 - Overflow and Carry Conditions, 126
 - Other Bases, 127
 - Summary of Rules for Complementary Numbers, 128
- 5.3** Real Numbers, 128
 - A Review of Exponential Notation, 128
 - Floating Point Format, 130
 - Normalization and Formatting of Floating Point Numbers, 133
 - A Programming Example, 135
 - Floating Point Calculations, 136
 - Floating Point in the Computer, 139
 - Conversion between Base 10 and Base 2, 141
- 5.4** Programming Considerations, 142
 - Summary and Review, 143
 - For Further Reading, 144
 - Key Concepts and Terms, 144
 - Reading Review Questions, 144
 - Exercises, 145

PART III COMPUTER ARCHITECTURE AND HARDWARE OPERATION

6 THE LITTLE MAN COMPUTER 152

- 6.0** Introduction, 153
- 6.1** Layout of the Little Man Computer, 154
- 6.2** Operation of the LMC, 154
- 6.3** A Simple Program, 156
- 6.4** An Extended Instruction Set, 157
- 6.5** The Instruction Cycle, 160
- 6.6** A Note Regarding Computer Architectures, 162
- Summary and Review, 163
- Key Concepts and Terms, 163
- Reading Review Questions, 164
- Exercises, 164

7 THE CPU AND MEMORY 166

- 7.0** Introduction, 167
- 7.1** The Components of the CPU, 168
- 7.2** The Concept of Registers, 169

- 7.3** The Memory Unit, 171
 - The Operation of Memory, 171
 - Memory Capacity and Addressing Limitations, 175
 - Primary Memory Characteristics and Implementation, 176
- 7.4** The Fetch–Execute Instruction Cycle, 178
- 7.5** Classification of Instructions, 181
 - Data Movement Instructions (LOAD, STORE, and Other Moves), 183
 - Arithmetic Instructions, 183
 - Boolean Logic Instructions, 184
 - Single Operand Manipulation Instructions, 184
 - Bit Manipulation Instructions, 184
 - Shift and Rotate Instructions, 185
 - Program Control Instructions, 186
 - Stack Instructions, 187
 - Multiple Data Instructions, 189
 - Other Instructions, 190
- 7.6** Instruction Word Formats, 190
- 7.7** Instruction Word Requirements and Constraints, 192
- Summary and Review, 195
- For Further Reading, 195
- Key Concepts and Terms, 195
- Reading Review Questions, 196
- Exercises, 197

8 CPU AND MEMORY: DESIGN, ENHANCEMENT, AND IMPLEMENTATION 200

- 8.0** Introduction, 201
- 8.1** CPU Architectures, 202
 - Overview, 202
 - Traditional Modern Architectures, 202
- 8.2** CPU Features and Enhancements, 203
 - Introduction, 203
 - Fetch–Execute Cycle Timing Issues, 204
 - A Model for Improved CPU Performance, 206
 - Scalar and Superscalar Processor Organization, 210
- 8.3** Memory Enhancements, 213
 - Wide Path Memory Access, 214
 - Memory Interleaving, 214
 - Cache Memory, 215
- 8.4** The Compleat Modern Superscalar CPU, 219
- 8.5** Multiprocessing, 221
- 8.6** A Few Comments on Implementation, 225
- Summary and Review, 225
- For Further Reading, 226

Key Concepts and Terms, 227
Reading Review Questions, 227
Exercises, 228

9 INPUT/OUTPUT AND BUSES 230

9.0 Introduction, 231
9.1 Characteristics of Typical I/O Devices, 232
9.2 Programmed I/O, 237
9.3 Interrupts, 239
 Servicing Interrupts, 239
 The Uses of Interrupts, 241
 Multiple Interrupts and Prioritization, 245
9.4 Direct Memory Access, 249
9.5 I/O Controllers, 252
9.6 Buses, 254
 Bus Characteristics, 254
Summary and Review, 258
For Further Reading, 259
Key Concepts and Terms, 259
Reading Review Questions, 259
Exercises, 260

10 COMPUTER PERIPHERALS 262

10.0 Introduction, 263
10.1 The Hierarchy of Storage, 264
10.2 Solid-State Memory, 266
10.3 Magnetic Disks, 267
 Disk Arrays, 272
10.4 Optical Disk Storage, 274
10.5 Magnetic Tape, 276
10.6 Displays, 277
 Basic Display Design, 277
 Graphical Processing Units (GPUs), 279
 Liquid Crystal Display Technology, 282
 OLED Display Technology, 283
10.7 Printers, 284
 Laser Printers, 285
 Inkjet Printers, 285
10.8 User Input Devices, 286
 Keyboards, 287
 Pointing Devices, 287
 Alternative Sources of Alphanumeric Input, 288
 Scanners, 291

- Multimedia Input, 291
- Mobile Devices, 292
- 10.9** Network Communication Devices, 293
- Summary and Review, 293
- For Further Reading, 294
- Key Concepts and Terms, 294
- Reading Review Questions, 295
- Exercises, 295

11 COMPUTER SYSTEM ORGANIZATION 298

- 11.0** Introduction, 299
- 11.1** Putting the Pieces Together, 300
- 11.2** System Architecture, 305
 - Basic System Interconnection Requirements, 305
 - Bus I/O, 307
 - Channel Architecture, 311
 - Blurring the Line, 313
- Summary and Review, 313
- For Further Reading, 313
- Key Concepts and Terms, 314
- Reading Review Questions, 314
- Exercises, 314

PART IV NETWORKS, DATA COMMUNICATIONS, AND NETWORKED COMPUTER SYSTEMS

12 NETWORKS AND DATA COMMUNICATIONS—AN OVERVIEW 318

- 12.0** Introduction, 319
- 12.1** The Impact of Networking on Business Processes and User Access
to Knowledge and Services, 320
- 12.2** A Simple View of Data Communications, 321
- 12.3** Basic Data Communication Concepts, 324
 - Messages, 324
 - Packets, 325
 - General Channel Characteristics, 326
- 12.4** Networks, 330
 - Network Topology, 330
 - Types of Networks, 334
 - Network Interconnection, 347
- 12.5** Standards, 351
- Summary and Review, 352
- For Further Reading, 352
- Key Concepts and Terms, 352
- Reading Review Questions, 353
- Exercises, 353

13 ETHERNET AND TCP/IP NETWORKING 356

- 13.0** Introduction, 357
- 13.1** TCP/IP, OSI, and Other Communication Protocol Models, 357
- 13.2** Program Applications versus Network Applications, 362
- 13.3** The Physical and Data Link Layers, 362
 - The Physical Layer, 363
 - The Data Link Layer, 364
 - Hub-Based Ethernet, 366
 - Switched Ethernet, 366
 - Wireless Ethernet (Wi-Fi), 367
- 13.4** The Network Layer, 368
- 13.5** The Transport Layer, 372
- 13.6** IP Addresses, 376
 - IPv4 and DHCP, 376
 - IPv6, 379
- 13.7** Domain Names and DNS Services, 380
- 13.8** Quality of Service, 385
- 13.9** Network Security, 386
 - Physical and Logical Access Restriction, 386
 - Encryption, 387
- 13.10** Alternative Protocols, 387
 - A Comparison of TCP/IP and OSI, 388
 - Other Protocol Suites and Components, 388
 - SCSI Over IP, 389
 - Cellular Technology, 389
 - MPLS, 390
- Summary and Review, 390
- For Further Reading, 391
- Key Concepts and Terms, 391
- Reading Review Questions, 392
- Exercises, 392

14 COMMUNICATION CHANNEL TECHNOLOGY 396

- 14.0** Introduction, 397
- 14.1** Communication Channel Technology, 398
- 14.2** The Fundamentals of Signaling Technology, 400
 - Analog Signaling, 401
 - Digital Signaling, 411
 - Modems, 416
 - Two Examples of Alternative Advanced Technologies, 416
- 14.3** Transmission Media and Signaling Methods, 418
- 14.4** Wireless Technologies, 420
 - Cellular Technology, 421
 - Wi-Fi, 421
 - Bluetooth, 421

Summary and Review, 422
For Further Reading, 423
Key Concepts and Terms, 423
Reading Review Questions, 423
Exercises, 424

15 MODERN NETWORKED COMPUTER SYSTEMS 426

15.0 Introduction, 427
15.1 Distributed Systems, 428
15.2 Client–Server Computing, 430
15.3 Web-Based Computing, 433
15.4 Peer-to-Peer Computing, 435
15.5 Clusters, 436
 Overview, 436
 Classification and Configuration, 437
 Beowulf Clusters, 438
15.6 Storage Area Networks, 440
15.7 Cloud Computing, 441
15.8 Supercomputing, 443
 Grid Computing, 444
15.9 Networked Computer System Architecture Examples, 444
 Google: A System Architecture Example, 445
 Another Example: Facebook’s Application Architecture, 448
Summary and Review, 449
For Further Reading, 450
Key Concepts and Terms, 450
Reading Review Questions, 450
Exercises, 451

PART V THE SOFTWARE COMPONENT

16 OPERATING SYSTEMS: AN OVERVIEW 454

16.0 Introduction, 455
16.1 The Barebones Computer System, 456
16.2 The Operating Systems Concept: An Introduction, 457
16.3 Services and Facilities, 463
 User Interface and Command Execution Services, 464
 File Management, 465
 Input/Output Services, 466
 Process Control Management, 467
 Memory Management, 468
 Scheduling and Dispatch, 468
 Secondary Storage Management, 471
 Network and Communications Support Services, 471

Security and Protection Services, 472

System Administration Support, 473

16.4 Organization, 476

16.5 Types of Computer Systems, 479

Summary and Review, 483

For Further Reading, 483

Key Concepts and Terms, 483

Reading Review Questions, 484

Exercises, 484

17 THE USER VIEW OF OPERATING SYSTEMS 486

17.0 Introduction, 487

17.1 Purpose of the User Interface, 488

17.2 User Functions and Program Services, 490

Program Execution, 490

File Commands, 491

Disk and Other I/O Device Commands, 492

Security and Data Integrity Protection, 492

Interuser Communication and Data Sharing Operations, 493

System Status Information and User Administration, 494

Program Services, 495

17.3 Types of User Interface, 495

The Command Line Interface, 496

Batch System Commands, 498

Graphical User Interfaces, 499

Touchless Gesture- and Voice-Based Interfaces, 504

Trade-Offs in the User Interface, 505

Software Considerations, 506

17.4 X Window and Other Graphics Display Methodologies, 507

17.5 Command and Scripting Languages, 510

The Elements of a Command Language, 512

The Command Language Start-Up Sequence Files, 512

17.6 Services to Programs, 513

Summary and Review, 515

For Further Reading, 515

Key Concepts and Terms, 515

Reading Review Questions, 516

Exercises, 516

18 FILE MANAGEMENT 518

18.0 Introduction, 519

18.1 The Logical and Physical View of Files, 519

18.2 The Role of the File Management System, 524

18.3 Logical File Access Methods, 528

Sequential File Access, 529

- Random Access, 529
- Indexed Access, 530
- 18.4** Physical File Storage, 531
 - Contiguous Storage Allocation, 531
 - Noncontiguous Storage Allocation, 532
 - Indexed Allocation, 534
 - Free Space Management, 537
 - Tape Allocation, 539
 - Optical and Flash Drive Allocation, 539
- 18.5** File Systems, Volumes, Disks, Partitions, and Storage Pools, 539
- 18.6** The Directory Structure, 542
 - Tree-Structured Directories, 543
 - Acyclic-Graph Directories, 545
- 18.7** Network File Access, 548
- 18.8** File Protection, 550
- 18.9** Journaling File Systems, 551
- Summary and Review, 552
- For Further Reading, 552
- Key Concepts and Terms, 553
- Reading Review Questions, 553
- Exercises, 554

19 THE INTERNAL OPERATING SYSTEM 556

- 19.0** Introduction, 557
- 19.1** Fundamental OS Requirements, 558
 - Example: A Simple Multitasking Operating System, 559
- 19.2** Starting the Computer System: The Bootstrap, 562
- 19.3** Processes and Threads, 564
 - Process Creation, 567
 - Process States, 568
 - Threads, 569
- 19.4** Basic Loading and Execution Operations, 570
- 19.5** CPU Scheduling and Dispatching, 572
 - Long-Term Scheduler, 572
 - Short-Term Scheduler and Dispatcher, 572
 - Nonpreemptive Scheduling Algorithms, 575
 - Preemptive Short-Term Scheduling Algorithms, 576
- 19.6** Memory Management, 577
 - Memory Partitioning, 578
- 19.7** Virtual Storage, 579
 - Overview, 579
 - Pages and Frames, 580
 - The Concept of Virtual Storage, 585
 - Page Faults, 586
 - Working Sets and the Concept of Locality, 588
 - Page Sharing, 588
 - Page Replacement Algorithms, 589

- Thrashing, 592
- Page Table Implementation, 592
- Segmentation, 595
- Process Separation, 596
- 19.8** Disk Scheduling, 596
 - First-Come, First-Served Scheduling, 597
 - Shortest Distance First Scheduling, 597
 - Scan Scheduling, 597
 - n*-Step c-Scan Scheduling, 598
- 19.9** Network Operating System Services, 598
 - OS Protocol Support and Other Services, 598
- 19.10** Other Operating System Issues, 601
 - Deadlock, 601
 - Other Issues, 602
- 19.11** Virtual Machines, 603
- Summary and Review, 605
- For Further Reading, 606
- Key Concepts and Terms, 606
- Reading Review Questions, 607
- Exercises, 608

BIBLIOGRAPHY B-1

INDEX I-1

SUPPLEMENTARY CHAPTERS

On the Web at www.wiley.com/Englander6e

SUPPLEMENTARY CHAPTER 1 An Introduction To Digital Computer Logic

- S1.0** Introduction
- S1.1** Boolean Algebra
- S1.2** Gates and Combinatorial Logic
- S1.3** Sequential Logic Circuits
- Summary and Review
- For Further Reading
- Key Concepts and Terms
- Reading Review Questions
- Exercises

SUPPLEMENTARY CHAPTER 2 System Examples

- S2.0** Introduction
- S2.1** Hardware Examples
 - The x86 Family
 - The POWER Family
 - The IBM System 360/370/390/zSeries Family
- S2.2** Operating System Examples
 - The Microsoft Windows Family

UNIX and Linux
The IBM z/OS Operating System
S2.3 Networking Examples
Google
Summary and Review
For Further Reading
Key Concepts and Terms
Reading Review Questions
Exercises

SUPPLEMENTARY CHAPTER 3 Instruction Addressing Modes

S3.0 Introduction
S3.1 Register Addressing
S3.2 Alternatives to Absolute Addressing
S3.3 Alternatives to Direct Addressing
Immediate Addressing
Indirect Addressing
Register Indirect Addressing
Indexed Addressing
Indirect Indexed and Indirect Indexed Addressing
Summary and Review
For Further Reading
Key Concepts and Terms
Reading Review Questions
Exercises

SUPPLEMENTARY CHAPTER 4 Programming Tools

S4.0 Introduction
S4.1 Program Editing and Entry
S4.2 The Concept of Program Translation
S4.3 Assembly Language and the Assembler
Operation of the Assembler
Assembly Language Formats
Features and Extensions
Relocatability
S4.4 Program Language Description and Rules
A Description of Written English
Programming Language Rules
Computer Language Descriptions
The Compilation Process
Interpreters
S4.5 Linking and Loading
S4.6 Debuggers
Summary and Review
For Further Reading
Key Concepts and Terms
Reading Review Questions
Exercises

PREFACE

The modern world offers a lot of readily available online resources for learning. Wikipedia, Google, news sources, millions of websites and blogs, even YouTube, offer access to information in nearly any subject that triggers your curiosity and interest. Nonetheless, we continue to believe that for deep understanding of something, nothing beats the integrated approach and focus of an old-fashioned printed-on-paper textbook. Well—maybe the e-book equivalent, but, still, a textbook.

When you open a new book, in *any* subject, the first thing you would usually want to know is what the book has to offer that makes it worth your while to read it. We would like to try to help you answer that question for the book that you're holding in your hand or on your tablet.

The information systems and technology fields are wonderfully exciting places to be! It seems as though every day brings new developments that alter the ways we create and work with information. Of course, with this excitement comes a challenge. To be a successful player in IS or IT, we have to be adaptable and flexible.

Much of the change occurs around computer system technology. The computer is, after all, at the foundation of information systems. A deep understanding of computer systems is, therefore, an essential element of success. We must be able to understand each new development, assess its value, and place it in the context of our knowledge of computer systems.

The subject of this book is the architecture of computer systems. Computer architecture is about the structure and operation of digital computers and computer-based devices. Computer architecture is concerned with the operational methods of the hardware; with the services provided by operating system software; with the acquisition, processing, storage, and output of data; and with the interaction between computer-based devices.

There is a tendency for people in information systems and technology to neglect a study of computer architecture. After all, the technology changes so rapidly—is it really worth trying to understand something that may be out of date by the time you finish this book? There is no question that computer technology has evolved rapidly. The computer in a smartphone is far more powerful than the mainframe computer of twenty-five years ago, with memory, disk and flash storage capacity, display and multimedia capability, and ease of use that would have been unthinkable just a few years ago. Even more important, connecting systems to work together is now routine and simple.

Interestingly enough, however, as profound as advances in the technology have been, the concepts of computer architecture that really matter have changed only nominally over the last seventy or eighty years. The new technologies are based on a foundation of architectural concepts that were developed many years ago. The architecture of a modern computer system was developed in the 1940s. The instruction set in a modern personal computer or smartphone is nearly identical to that of computers built in the 1950s and 1960s. Modern operating system techniques were developed in the 1960s. The graphical user interface is based on a 1960s' project. The Internet is built from concepts developed more than fifty years ago.

So you see that an understanding of computer architecture makes it possible to “ride the wave” of technological change, secure in the feeling that you are equipped to deal with new

developments as they occur, and to have fun doing so. When you are done reading this book, you will have substantial knowledge about how a computer works and a good understanding of the operating concepts, the hardware, and system software that make up a computer. You will see the interaction between computers and between data and the computer. Plus, you will have learned lots of jargon that you can show off at parties and job interviews.

This textbook is designed for a wide range of readers, both undergraduate and graduate. The material is specifically directed toward IS and IT majors. There are no explicit prerequisites, although the book assumes that the student is familiar with a personal computer. It also assumes (but does not require) some basic programming skills: although there is no programming in the book, program code is occasionally used as an example to clarify an idea, and a knowledge of programming is helpful at understanding instruction set design and program execution concepts. The material in this textbook conforms to the criteria of the IT infrastructure courses and core concepts, as described in the ACM and AIS standard IS2010 and IT2017 curricula, together with proposed updates in 2020. Although the material in this book may be useful as background for system design and implementation project courses, the course can be placed anywhere in the curriculum.

Most instructors will not cover the entire textbook in a single semester. The organization of this book is designed to allow an instructor to cover the major topic areas in different levels of depth, depending on the experience and needs of the students. On the other hand, it is my intention that this book will serve a student as a useful reference long after the formal course is completed. It is designed for use as a book where a professional can look up the basic concepts that clarify new developments as they occur.

This text is the outgrowth of courses that Professor Englander taught to CIS majors and minors at Bentley University at both the undergraduate and graduate levels for more than thirty-five years. Professor Wong joined the Bentley CIS Department as a colleague in 2001 and is now teaching in the Computer Science department at Worcester Polytechnic Institute. Student responses to the material and the approach have generally been very enthusiastic. Many students have returned after graduation to tell us that their knowledge in this area has directly contributed to their career development. Along the way, student comments have also been extremely helpful to us in the book's continuing evolution.

Those familiar with previous editions will notice that the organization of recent editions has undergone continual update and revision to reflect current technological practices and trends. In particular, the sixth edition reflects the trend toward universal connectivity among a wide variety of computer-based devices. Thus, there is an increased emphasis on the integration and synergy of the various components of the computer system and on the system concept as a whole. Still, the basic philosophy, organization, and approach remain essentially similar to those of the first edition, reflecting the unchanging nature of the underlying principles.

Organization of the Sixth Edition, Noting Changes from Previous Editions

The biggest challenge for us as the authors of this book has been to preserve the guiding principles established in the first edition, while reflecting the major changes in the way computers are used, in the rapid deployment of new technology, and in the resulting evolution of IS/IT curriculum to reflect those changes. The sixth edition is a substantial, though incremental, update to previous editions. It reflects the rapid growth in the use of tablets and smartphones as important computing devices. The material on networking has been substantially reorganized. Still, users of previous editions will find that much of the material is familiar; after all, the way in which computers are used in IS/IT may have changed, but

the basic guiding principles of computer architecture are essentially the same as they have been for many years.

This book is organized into five parts totaling nineteen chapters, plus four additional supplementary chapters that are posted on the Web. The first part serves as an introduction and overview of the role of the computer in information systems; it introduces the concept of a system and provides a brief introduction to each of the components that make up a modern computer system. Each of the remaining four parts deals with a single architectural aspect of the computer system.

Part 2 discusses the role and representation of data in the computer. Here, we consider numbers, text, sound, images, video, and other data forms. Part 3 presents the hardware architecture and operational concepts. It introduces the components of a computer and shows how they collaborate to execute computer instructions, discusses the nature of a computer instruction set, and explores the interaction between the CPU, memory, and I/O peripheral devices. Part 4 presents a thorough introduction to the basics of computer networking and culminates with a survey of the major types of modern networked computer systems. Part 5 discusses the system software, the programs that function to make the resources of the computer system, and other interconnected computer systems and components, accessible to the user and to application programs.

The approach within each group of chapters is layered. Each new layer builds upon the previous material to add depth and understanding to the reader's knowledge. Each topic section consists of a short introduction that places the topic to be discussed into the context of the computer system as a whole and then lays out in detail the organization of the chapters within the section. Each topic area is introduced as gently as possible, using ideas and examples that are already familiar to students. In addition to the numerous examples that are used throughout the text, the supplementary chapters offer substantial case studies that show application of the text material to current examples of importance. Overall, the approach is gentle, progressive, and accumulative. As much as possible, each section is self-contained.

An overview of the organization of each part follows. A few comments about the differences between the fifth and sixth editions are included for those who are familiar with the previous edition. More details can be found in the introductions to each section.

Part 1 consists of two chapters that present a short overview of computing and place architectural concepts into the context of information technology. Chapter 1 introduces the components of a computer system and shows the relationships among the components. In the new edition, we have assumed that today's students are more familiar with computer technology and terminology than previous generations, so there is less introduction of "IT101"-type material than that found in earlier editions. Chapter 1 also presents a simple model of computing and discusses the importance of standards and protocols in the development of computer systems. This chapter concludes with a short history of computers from the architectural point of view. Chapter 2 focuses on the concepts of systems, models, and a brief introduction to system architectures. The subsections of Section 2.2 dealing with distributed processing systems and their accompanying examples have been moved to a new Chapter 15 that gathers together all of the related topics on modern networked computing systems.

Chapters 3 through 5 comprise Part 2. Chapter 3 introduces number systems and basic number system operations; it then explores the relationships between numbers in different number bases and the conversion techniques between the different representations. Hexadecimal numbers are now treated as just another base. As a result, the treatment of hexadecimal numbers in Section 3.5 has been distributed among the other chapter sections. Chapter 4 investigates different types of data formats, including alphanumeric, image,

video, and audio formats. It considers the relationship between numerical and character-based representations. Chapter 5 studies the various formats that are used to represent and to perform calculations on integer and floating point numbers.

Part 3 discusses the hardware architecture and operational aspects of the computer. Chapter 6 begins the study with the introduction of the Little Man Computer, a simple model that provides a surprisingly accurate representation of the CPU and memory. The model is used to develop the concept of an instruction set and to explain the basic principles of the von Neumann architecture. Chapter 7 extends the discussion to a real computer. It introduces the components of the CPU and shows their relationship to the Little Man Computer model. It explains the operation of memory, presents the instruction fetch–execute cycle, and discusses the instruction set. It identifies important classes of instructions and discusses the ways in which instructions can be categorized. The ARM instruction set is offered as an illustration of a typical current CPU model. Discussion of buses has been moved to Chapter 9, “Input/Output and Buses”.

Chapter 8 expands the material in Chapter 7 to consider more advanced features of the CPU and memory. It offers an overview of various CPU architectures. It continues with a discussion of techniques for improving memory access, particularly cache memory, and an introduction to current CPU organization, design, and implementation techniques, including pipelining and superscalar processing. This chapter also introduces multiprocessing (or multicore, in current terminology) concepts.

Chapter 9 presents the principles of I/O operation, including a discussion of buses, which was previously found in Chapter 7. Chapter 10 illustrates how I/O is performed in various I/O devices.

Chapter 11 views the computer system as a whole. It discusses the interconnection techniques and integration of the various hardware components. Chapter 11 examines modern systems, including mobile systems, and System-on-a-Chip concepts. Topics in Chapter 11 that addressed the interconnection of computers to increase the performance and reliability of a computer system such as clustering, supercomputing, and grid computing have been moved to a new Chapter 15 that unifies discussion of networked computing systems.

Three supplementary chapters on the Web provide additional resources to support the chapters in Part 3. Supplementary Chapter 1 (SC1) offers an introduction to Boolean algebra, combinatorial logic, and sequential logic for those readers that would like a deeper understanding of the computer in its simplest and most elegant form. Supplementary Chapter 2 (SC2) offers three detailed case studies of important architectures: the Intel x86 family, the POWER computer, and the IBM zSystem. As of this writing, this supplement is currently being updated. Supplementary Chapter 3 (SC3) discusses alternative approaches to instruction addressing.

Part 4, Chapters 12–15, presents a thorough introduction to networking.

Chapter 12 introduces the concept of a communication channel and explores its characteristics and configurations. This includes the communication channel model, the concept of links, packets, basic channel characteristics, network topology, types of networks (LAN, MAN, etc.), and basic network interconnection and routing.

Chapter 13 focuses on the passage of data packets through a network, concentrating on Ethernet and TCP/IP network technologies. Section 13.1 introduces the concept of layered communication, utilizing both the TCP/IP and OSI models. Section 13.2 describes the differences between program applications and network applications. Following these two sections, the next three sections carefully describe the process of moving packets, one layer at a time, from the bottom up. Sections 13.6 and 13.7 explain IPv4 and IPv6 addressing and DHCP, and DNS, respectively. Quality of Service and network security are briefly introduced

in Sections 13.8 and 13.9. The chapter concludes with a discussion of alternative protocols, including a comparison of OSI and TCP/IP, as well as brief discussions of MPLS and cellular technology. Frame Relay and SONET technologies have been removed from the chapter as faster Ethernet speeds have rendered these technologies obsolete.

Chapter 14 focuses primarily on communication channel technology, including analog and digital signaling, modulation and data conversion techniques between analog and digital, and the characteristics of transmission media.

Combining modern computer systems with network technologies, Chapter 15 is a new chapter that aims to present a unified and systematic exploration of the different types of networked computer systems. These systems include traditional ones such as clusters, client-server computing, peer-to-peer computing, supercomputing, and grid computing, as well as newer networked systems such as storage area networks (SANs), Web-based computing, and cloud computing. The chapter also includes examples of two modern networked architectures, those of Google and Facebook, previously found in Chapter 2.

Part 5 is dedicated to a discussion of system software. Chapter 16 provides an overview of the operating system. It explains the different roles played by the operating system and introduces the facilities and services provided. Chapter 17 presents the role of the operating system from the viewpoint of the user of a system. The sixth edition offers new screenshots of Windows 10, macOS, and recent versions of iPhone and Android mobile phones. Chapter 18 discusses the all-important topic of file systems. Chapter 19 discusses the operating system as a resource manager, with an in-depth discussion of memory management, scheduling, process control, network services, and other basic operating system services. The chapter also includes an introduction to virtual machines and hypervisors. In addition to its hardware discussions, Supplementary Chapter 2, when completed, will also provide current Windows, UNIX/Linux, and z/OS case studies.

A fourth supplementary chapter provides an introduction to the system development software that is used for the preparation and execution of programs.

This book has been a continuing labor of love. Our primary goal has been to create and maintain a textbook that explains computer architecture in a way that conveys to you, the reader, the sense of excitement and fun that we believe makes a career in information systems and technology so satisfying. We hope that we have succeeded to some extent.

Additional Resources

Additional resources for students and instructors may be found at the textbook's website, www.wiley.com/Englander6e. We can also be reached directly by e-mail at ienglander@bentley.edu and wwong2@wpi.edu. Although we are happy to communicate with students, we are unable to supply tutorial help or answers to review questions and exercises in this book.

Acknowledgment and Note

Over the previous five editions and 25+ years, I've had the opportunity to acknowledge and thank dozens of colleagues, reviewers, error reporters, test bank and classroom slide presentation authors, folks offering ideas and suggestions for improvement, a developer of software to accompany the text, editors and production people at Wiley, and friends who have helped me make the effort to produce a useful and accurate textbook a pleasure. A major, ongoing textbook project is a formidable task. These individuals have helped to make the task manageable—and kept me going when, from time to time, I became convinced that textbooks really *do* appear by magic and are *not* written by humans. It is impossible to thank all these wonderful people adequately for all their help and support.

Now, one of those friends and colleagues, Professor Wilson Wong has joined me as coauthor of the sixth edition of this book. Wilson has previously taught the course that uses this textbook and contributed the first set of PowerPoint slides for the book to Wiley. I'm sure his technical expertise and insight will further improve understanding and access to the material in this textbook as this and future editions move forward.

As I move to thank the folks who made this edition possible, I wish to issue a special shout-out thank you to Beth Golub, my editor for the first five editions, who encouraged, supported, and cajoled me into making this book as useful as humanly possible. Continuing that trend, special kudos to Jennifer Manias, the current editor, Judy Howarth, the senior managing editor, Umamaheswari Gnanamani, the production editor, and all the others who have managed to put a book together in the middle of a pandemic. Thank you all!

I have strived to make this book as technically accurate as is humanly possible. Nonetheless, I know that errors have a way of creeping in when one least expects them. I would greatly appreciate hearing from readers who find errors that need correction. Your comments and suggestions about the book are also welcome.

IRV ENGLANDER

ienglander@bentley.edu
Boston, Massachusetts

ABOUT THE AUTHORS

Dr. Irv Englander has been involved in many different aspects of the computing field for more than sixty years. He has designed logic circuits, developed integrated circuits, developed computer architectures, designed computer-controlled systems, designed operating systems, developed application software, created the initial system design for a large water-monitoring system, performed software auditing and verification of critical control software, and developed and specified hardware components and application software as a consultant for business systems large and small.

As an educator, he has contributed papers and given workshops on end-user computing, e-commerce, and on various aspects of IS and IT pedagogy. He was an invited contributor and reviewer for the IS-97 and IS-2002 information systems curricula, and continues to publish and take an interest in the technical infrastructure components of the IS/IT curriculum. He is actively involved in the application of new technology to information systems. Most recently he served as a faculty advisor to a low-cost supercomputing student research project, which was presented at the ACM/IEEE Computer Society Supercomputing Conference SC13.

Dr. Englander has a Ph.D. from MIT in Computer Science. His doctoral thesis was based on the design of a large image-processing software laboratory. At MIT, he won the Supervised Investors Award for outstanding teaching. He holds the rank of Professor Emeritus of Computer Information Systems at Bentley University, where he taught full-time for thirty-eight years.

Dr. Wilson Wong likes the fact that he was born in 1965, the same year the IBM 360 main-frame computer made its first appearance. Graduating with a B.S. from MIT in Computer Science and Engineering and an M.B.A. from Cornell University, he spent the first twenty years of his career as a software developer, a director of IT departments, and Chief Technologist at companies and startups in various industries, including publishing, law, Web development, technology, and early e-commerce.

Dr. Wong entered academia in 2001; he spent 14 years at Bentley University in the Computer Information Systems department, teaching a wide variety of courses including computer architecture, database, software project design and management, and Web development. Although it was not in his original plans, he also earned a Ph.D. in Business with a concentration in Information Systems from Bentley along the way.

As an educator, he has presented papers on IS pedagogy, particularly in IS capstone project innovations, and serves regularly as a reviewer for various IS journals and conferences.

Dr. Wong is currently an Assistant Teaching Professor of Computer Science at Worcester Polytechnic Institute where he enjoys teaching databases and a software engineering course that includes collaborative projects with industry and the federal government. He is also the CS Faculty Advisor to the WPI Fintech Project Center, where he manages technical senior projects in collaboration with various Wall Street and International banking firms.

AN OVERVIEW OF COMPUTER SYSTEMS

A computer-based information system is made up of a number of different elements:

- The *data* element. Data is the fundamental representation of facts and observations. Data is processed by a computer system to provide the information that is the very reason for the computer's existence. As you know, data can take on a number of different forms: numbers, text, images, video, and sounds. But it's all numbers in the computer.
- The *hardware* element. Computer hardware processes the data by executing instructions, storing data, and moving data and information between the various input and output (I/O) devices that make the system and the information accessible to the users.
- The *software* element. Software consists of the system and application programs that define the instructions that are executed by the hardware. The software determines the work to be performed and controls operation of the system.
- The *communication* element. Modern computer information systems depend on the ability to share processing operations and data among different computers and users, located both locally and remotely. Data communication provides this capability.

The combination of hardware, software, communication, and data make up the *architecture* of a computer system. The architecture of computer systems is remarkably similar whether the system is a personal laptop computer, a mobile phone, a PlayStation, an embedded computer that controls the functions in your car, or a large mainframe system that is never actually seen by the hundreds of users who access it every day.

Amazingly, the changes in computer technology that you've seen in just the last few years are essentially superficial; the basic architecture of computer systems has changed surprisingly little over the last sixty years. The latest IBM mainframe computer executes essentially the same instruction set as the mainframe computer of 1965. The basic communication techniques used in today's systems were developed in the 1970s. As new as it might seem, the Internet celebrated its fiftieth anniversary in 2020. All of this is surprising considering the growth of computing, the rapid change of technology, and the increased performance, functionality, ubiquity, and ease of use of today's systems. This makes the study of computer architecture extremely valuable as a foundation upon which to understand new developments in computing as they occur.

Computer system architecture is the subject of this textbook. Each element of the system is addressed in its own section of the text, always with an eye to the system as a whole.

Part 1 is made up of two chapters that present an overview of systems and of the computer system in particular.

Chapter 1 addresses a number of issues, including:

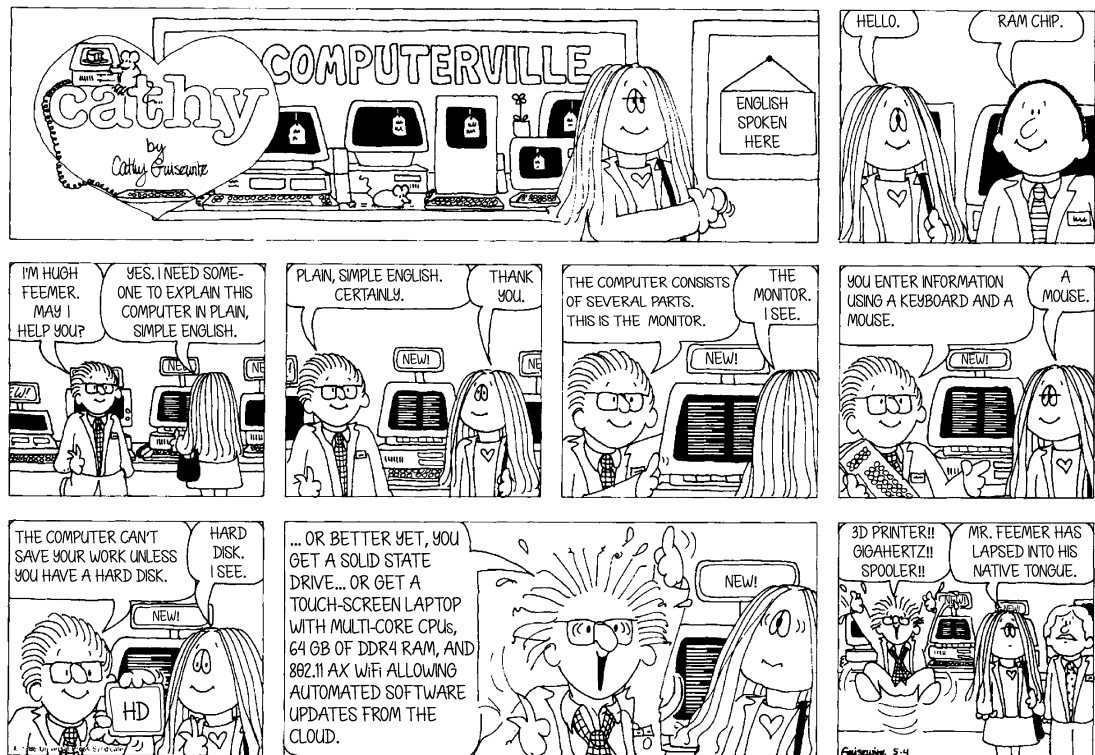
- The ways in which a knowledge of computer architecture enhances our abilities as computer users and professionals
- A simplified view of typical computer system architectures
- The basic components that make up a computer system
- The fundamental operations that are performed by computer systems.

Chapter 1 concludes with a brief architectural history of the computer.

An encompassing theme throughout this text is that of systems and system architecture. The words “system” and “architecture” appear throughout this book: we talk about information systems, computer systems, operating systems, file systems, software architecture, I/O architecture, network architecture, and more. You will probably take a course in *System Analysis and Design* sometime in your college career.

Although most people have an instinctive understanding of what a system is, it is more important for us as system professionals to understand the concepts of systems and system architecture at a deeper level than the average person. Chapter 2 offers careful definitions and examples of the concept of systems, both generally and in the specific context of the computer systems that are the focus of this book. A brief introduction to system architecture is presented at the end of Chapter 2, and discussed in much greater detail in Chapters 11 and 15.

Computers and Systems



CATHY © 1986 Cathy Guisewite. Reprinted with permission of UNIVERSAL PRESS SYNDICATE. All rights reserved.

1.0 Introduction

Welcome to the wonderful modern world of computers. Where the technology changes daily. Or does it?

It's true that the computer of today looks nothing like the computer of even five or ten years ago. It is nearly impossible today to escape the immediate and ubiquitous reach of computers and computer-based systems and devices. There is probably a smartphone in your pocket or on your desk. For many of you, your laptop or tablet is sitting nearby as you read this paragraph. (Or, maybe you're even *reading* this paragraph on your tablet or e-book.) Furthermore, your smartphone probably has more computing power than most of the computers on the market ten years ago. It fits easily in your pocket or purse. It weighs less than 1/100 of that old desktop computer and has at least ten times as much memory!

And that's not all. Your car has several embedded computers controlling various automotive functions, and perhaps a touch screen for hands-free telephone, navigation, the radio, Internet access, and more. Which is almost unnecessary, because you can probably tell it what you want verbally anyway. Even your microwave oven and the machine that launders your clothes depend on computers to function. As you are likely aware, most of these machines can talk to each other, using Bluetooth, the Internet, or some other networking technology. (Think IoT, the Internet of Things, as just one example.) Just for fun, Figure 1.1 shows pictures typical of a recent laptop, a 2020 smartphone, and a current embedded computer that controls many functions in your car.

Although the focus of this book is on IT systems, our discussion of computer hardware, software, and networking applies equally well to workplace computers, tablets, smartphones, and, even computers embedded in other devices and equipment. In this figure, we have three seemingly very different looking pieces of equipment working on different types of applications. And yet, it's hopefully obvious to you that these three systems share a lot in common. They are all computer based. All contain at least one central processing unit (CPU, some contain more) and memory. All provide a facility for interacting with long-term storage and other devices and with users. What may be less obvious to you is that the programs that they run are also essentially similar, differing mostly in the details required by the different components of the particular system and by the nature of the applications. For example, systems may have different amounts of memory, different types of displays, different I/O devices, and different operating systems, as well as running different types of applications and serving different purposes.

In fact, a modern IT system may contain elements of many different types of systems, with networking that ties everything together.

When creating an IT system, our concerns are whether the various components provide the features and performance that the users require. To be an effective designer and user, you have to understand the specifications, their importance and their meaning; the terminology; and the jargon. Which features are important to the users? Is this the right combination



Global American, Inc.

FIGURE 1.1
Computer Devices,
Old and New

of features that you need in your computer to have the computer perform the work that you wish to get done? Are there features missing that we need? Perhaps we are paying too much for the performance that we need. Or maybe we need more. What other information about this system would allow you to make a more informed decision? There is obviously no need to understand the inner workings of most modern computer-based systems to operate them adequately. Indeed, in many cases, the presence of the computer is hidden from us, or **embedded**, and its operation is invisible to us as users. We don't need to know how a computer works to read an e-book.

Even as experienced users, we can run standard software packages on a personal computer or apps on a smartphone without understanding exactly how they work; we can program a computer in a high-level or a scripting language without understanding the details of how the machine executes the individual instructions; we can design and implement Web pages without understanding how the Web browser gets its pages from a Web server or how the Web server creates those pages; we can purchase a tablet or a laptop computer from a salesperson without understanding the specifications of the system.

And yet, there is something missing. Perhaps the software doesn't do exactly what we want, and we don't understand the machine well enough to risk fooling around with the software's options. Perhaps if we understood the system better, we might have written and configured the program to be faster and more efficient. Perhaps we could create Web pages that load faster and work better. Perhaps the salesperson did not sell us the optimum system for our job. Or perhaps it's nothing more than a sense of excitement that's missing. But that's important, too!

You are reading this book because you are a student studying to become a computer professional, or maybe you are simply a user wanting a deeper understanding of what the computer is all about. In either case, you know you'll be interacting with computer systems in some form or other for the rest of your life. It's nice (as well as useful) to know something about the tools of the trade. More importantly, understanding the computer system's operations has an immediate benefit: it will allow you to use the machine more effectively.

As a user, you will be more aware of the capabilities, strengths, and limitations of a computer system. You will have a better understanding of the commands that you use. You will understand what is taking place during the operation of the program applications that you use. You will be able to make better informed decisions about your computer equipment and application programs. You will understand more clearly what an operating system is, and how to use it effectively and to your advantage. You will know when it is preferable to do a job manually, and when the computer should be used. You will understand the most efficient way to "go online", and what benefits might be gained from your home network. You will improve your ability to communicate with system analysts, programmers, and other computer specialists. You might even save money! Do you really need to pay for 50 GB of download data on your mobile phone every month? And what is that 50 GB being used for?

As a programmer, it will allow you to write better programs. You will be able to use the characteristics of the machine to make your programs operate more effectively. For example, choosing the appropriate data type for a variable can result in significantly faster performance. Soon you will know why this is so, and how to make the appropriate choices.

You will discover that some computers will process nested loops much more quickly if the index variables are reversed. A rather surprising idea, perhaps, and you'll understand why this is true.

You will understand why programs written in a compiled language like C++ usually run much faster than those written in interpreted program languages like BASIC or scripting languages like JavaScript. Similarly, you'll see why the basic layout of a program can have a major impact on the program's run-time efficiency.

As a systems architect or system analyst, you will be responsible for the design and implementation of systems that meet an organization's information technology (IT) needs, recognizing that the differences in the cost and capabilities of the components that you select may have significant impact on the organization. With the knowledge gained here you will be in a better position to determine and justify the set of computer system components and the system architecture that are appropriate for a particular job and to determine the trade-offs with other possible system architectures.

You'll be able to assist management in making intelligent decisions about system strategy: should the company adopt a large mainframe/virtual machine system approach for its Web servers, or would a system consisting of a network of off-the-shelf blade servers provide better performance at lower cost? You'll be better prepared to analyze the best way to provide appropriate facilities to meet the needs of your users. In an era of fast-changing technology, you'll be more able to differentiate between simple technological obsolescence that does not affect the organization's requirements significantly and major advances that suggest a real need to replace older equipment. You will understand the trade-offs inherent in the use of cloud and other remote services.

When selecting computers, you would like to purchase the computers and services that best meet the needs of the organization's applications and the users. You must be able to read and understand the technical specifications in order to compare different alternatives and to match the system to the users' needs. This book will teach you what you need to know to specify and purchase a system intelligently. You'll know the differences between various CPU technologies and the advantages and disadvantages of each. You will learn what peripheral hardware is appropriate for your organization's files and the trade-offs between different file system formats, what is required to build an intranet, and what the speed, size, and performance limitations of a particular system are. You'll be able to compare the features of macOS, Windows, Chrome OS, and Linux knowledgeably and decide which ones are important to you. You'll be able to apply your basic understanding of computers to new technologies and concepts such as mobile IT, new network protocols, virtual machines and cloud services as they appear. You'll learn to understand the jargon used by computer salespeople and judge the validity of their sales claims.

As a networking professional, you are responsible for the design, maintenance, support, and management of the networks that connect your computer systems together and provide the required interfaces to the outside world. You must be able to specify network layouts that optimize your equipment and network resources. Your understanding of basic network configurations and protocols will allow you to control and provide sufficient and appropriate access to your users in an efficient manner. This text offers the basic tools as a starting point to prepare for a career in networking.

As a Web services designer, you must be able to make intelligent decisions to optimize your Web system configurations, page designs, data formatting and scripting language choices, security features, and operating systems to optimize customer accessibility to your Web services.

As a system administrator or manager, your job is to maximize the availability and efficiency of your systems. You will need to understand the reports generated by your systems and be able to use the information in those reports to make changes to the systems that will optimize system performance. You will need to know when additional resources are required, and be able to specify appropriate choices. You will need to specify and configure operating system parameters, set up file systems, select cloud services, manage system and user PC upgrades in a fast-changing environment, reconfigure networks, provide and ensure the robustness of system security and data backup, and perform many other system management tasks. The configuration of large systems can be very challenging. This text will give you an understanding of operating system tools that is essential to the effective management of systems.

In brief, when you complete this book, you will understand what computer hardware and software are and how programs and data interact with the computer system. You will understand the computer hardware, software, and communication components that are required to make up a computer system and what the role of each component in the system is.

You will have a better understanding of what is happening inside the computer when you interact with the computer as a user. You will be able to write programs that are more efficient. You will be able to understand the function of the different components of the computer system and to specify the computer equipment and resources you need in a meaningful way. You will understand the options that you have as a system administrator or Web services or network designer.

In an era in which technology changes extremely rapidly, the architecture of the computer system rests on a solid foundation that has changed only slightly and gradually over the last sixty years. Understanding the foundations of computer system architecture makes it possible to flow comfortably with technological change and to understand changes in the context of the improvements that they make and the needs that they meet. In fact, interviews with former students and with IT executives and other IT professionals clearly indicate that a deep understanding of the basic concepts presented here is fundamental to long-term survival and growth in the field of information technology and IT management.

This type of understanding is at the very foundation of being a competent and successful system analyst, system architect, system administrator, or programmer. It may not be necessary to understand the workings of an automobile engine in order to drive a car, but you can bet that a top-notch race car driver knows his or her engine thoroughly and can use it to win races. Like the professional race car driver, it is our intention to help you to use your computer engine effectively to succeed in using your computer in a winning way. The typical end user might not care about how their computer system works, but you do.

These are the goals of this book. So let's get started!

1.1 The Starting Point

Before we begin our detailed study of the architecture of computer systems, let us briefly review some of the fundamental principles, characteristics, and requirements that guide computer system design and operation. The fundamentals described here apply to computers in general, regardless of size or purpose, from the smallest embedded device to the largest mainframe computer.

In a simple scenario, you use your tablet, laptop, or desktop personal computer to word process a document. You probably use a pointing device such as a mouse or stylus or finger to move around the document and to control the features of the word processor software application, and you use a keyboard or touch screen to enter and modify the document text data. The word processor application program, together with your document, appears on a screen. Ultimately, you might print the document on a printer. You store the document on a solid-state drive (SSD), disk, flash drive, or some other storage device.

The fundamentals of a typical computer system are readily exposed in this simple example. Your pointing device movements and clicks and your text data entry represent input to the system. The computer processes the input and provides output to the screen, and, perhaps, to a printer. The computer system also provides a storage medium of some sort, usually an SSD or a hard disk, to store the text for future access. In simplest terms, your computer receives input from you, processes it, and outputs results to the screen. Your input takes the form of commands and data. The commands and programs tell the computer how to process the data.

Now consider a second, slightly more complex example. Your task in this example is to access a Web page on the Internet. Again, your input to the computer is via keyboard and pointer control device. When you type the Web page URL, however, your computer sends a message to another computer that contains Web server software. That computer, in turn, sends a Web page file that is interpreted by the browser on your computer and presented on your screen. You are probably already aware that Hypertext Transfer Protocol (HTTP) is used as a standard for Web message exchanges.

The elements of this example differ only slightly from the first example. Your command inputs tell a Web browser software application on your computer what processing is to take place; in this case, your desire to access a Web page. The output from your computer is a message to a Web server on the remote computer requesting the data that represents the Web page. Your computer receives the data as input from the network; the Web browser processes the data and presents the Web page output on the screen. Figure 1.2 illustrates the layout for this example.¹

The major differences between this and the first example are the source of the input data and the fact that the second example requires network connectivity between the two computers. Instead of the keyboard, the input data to be processed by your Web browser comes from a communication channel. (Note that the exact nature of the channel is not important for this discussion.) In both cases, your computer receives data input to process, and control input, usually HTML or XML, that determines how the data is to be processed, performs the processing, and provides output.

These two examples contain all of the key elements found in any IT system, large or small.

- An IT system consists of one or more computer systems; multiple computer systems are connected together using some type of network interconnectivity. As a matter of necessity, network interfaces must conform to standard agreements, known as **protocols**, for messages to be understood by both computers during a message exchange between a pair of computers. The network itself can take on a variety of forms, provided that the interface requirements are met, and are determined by such characteristics as performance, convenience, and cost.

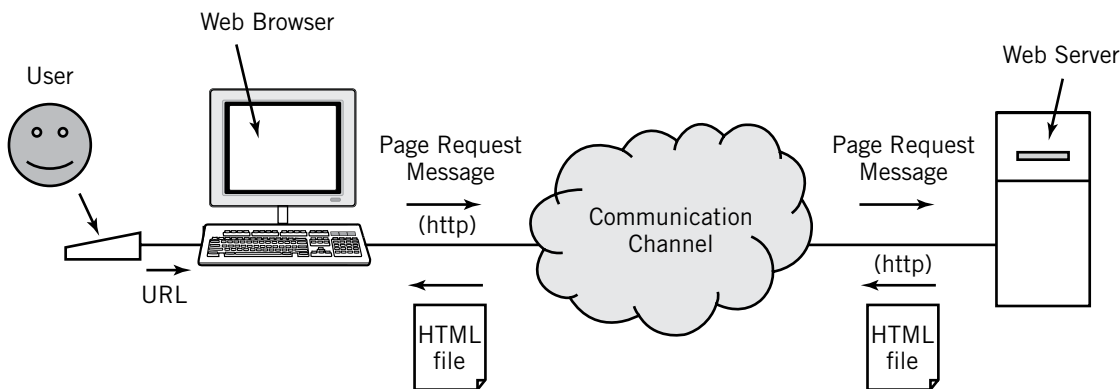
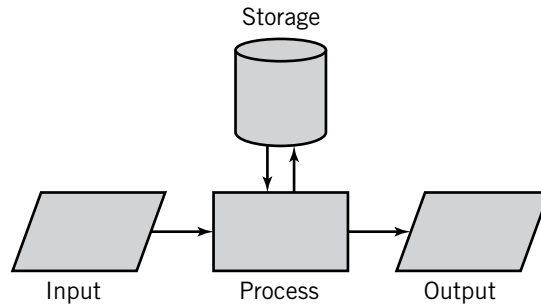


FIGURE 1.2 Typical Web Browser Application Use

¹The cloud in the image refers to the network connection that makes communication possible between the two computers, rather than the computing services delivered over the Internet known as cloud computing.

FIGURE 1.3 A
Computer Process



- The work performed by an individual computer system within the IT system can be characterized by input, processing, and output. This characterization is often represented by the **input–process–output (IPO) model** shown in Figure 1.3. Storage is also represented within this model. Alternatively, storage can be interpreted as output to be saved for use as future input. Storage is also used to hold the software programs that determine the processing operations to be performed. The ability to store programs and data on a temporary, short-term, or long-term basis is fundamental to the system. In Chapter 2, Section 2.2, and Chapter 15, we will show that all IT systems can ultimately be characterized by the same basic IPO model at all levels, from a single computer to a complex aggregation of computers, although the complexity of large systems may obscure the model and make it more difficult to determine the actual inputs, outputs, and processing operations. The IPO model provides an important basic tool for system analysis and design practices.
- The components of an individual computer system consist of processing hardware, input devices, output devices, storage devices, application software, and operating system software. The task of the operating system software is to provide overall control of the individual system, including management of input, output, and file storage functions. The medium of exchange, both with users and between computers within a larger system, is data. (Note that the messages between computers in the second example are a form of data.) Figure 1.4 is a simple illustration of computer systems embedded in a larger IT system.

Figure 1.5 summarizes the basic operations that are performed during computer processing. These operations, in turn, can be reduced to the primitive operations that are also familiar to you from your understanding of programming languages. The primitive processing operations common to high-level programming languages are shown in Figure 1.6.

1.2 Components of the Computer System

As noted in the previous section, there are three components required for the implementation of a computerized input–process–output model:

1. The computer hardware, which provides the physical mechanisms to input and output data, to manipulate and process data, and to electronically control the various input, output, and storage components.
2. The software, both application and system, which provides instructions that tell the hardware exactly what tasks are to be performed and in what order.

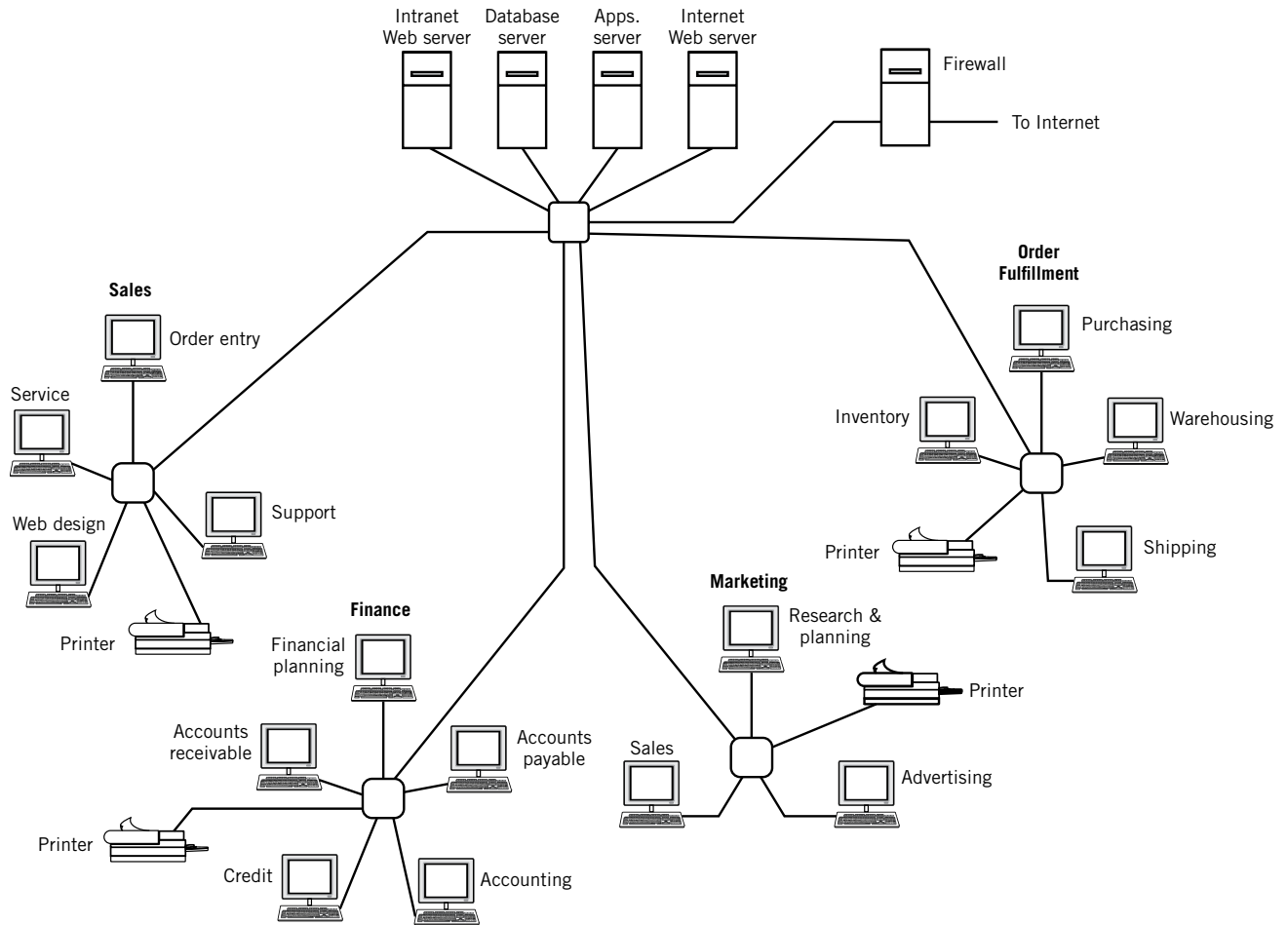


FIGURE 1.4 A Simplified IT Computer System Layout

- Input/output
- Basic arithmetic and logical calculations
- Data transformation or translation (e.g., program compilation, foreign language translation, file updating)
- Data sorting
- Searching for data matches
- Data storage and retrieval
- Data movement (e.g., movement of text or file data to make room for insertion of additional data)

FIGURE 1.5 Basic Computer Operations

- Input/output (including file storage and retrieval)
- Arithmetic and logical assignment statements
- True/false decision branching (IF-THEN-ELSE or IF-GOTO)
- Loops and/or unconditional branching (WHILE-DO, REPEAT-UNTIL, FOR, GOTO)

FIGURE 1.6 Basic High-Level Language Constructs

3. The data that is being manipulated and processed. This data may be numeric, it may be alphanumeric, it may be graphic, or it may take some other form, but in all cases, it must be representable in a form that the computer can manipulate.

In modern systems, input entry, output display, and storage of the data and software used for processing often take place at a location different from the computer where the actual processing occurs. In many installations, actual processing is distributed among computer systems, with particular results passed to the individual systems that require them. Therefore, we must also consider a fourth component:

4. The communication component, which consists of hardware and software that transport programs and data between interconnected computer systems.

The hardware and system software components make up the architecture of the computer system. The communication component connects individual computer systems together to form an organizational system architecture. The data component and also the application software, while fundamental to the operation of the computer system, are supplied to the computer system by the user or vendor, rather than being a part of the architecture of the computer system itself. (It is useful to note, however, that application software and data structure *are* often considered as part of the overall system architecture when one considers the architecture from the perspective of the organization. We explore this issue in Chapter 15, Section 15.3. Note, however, that the focus of this book is primarily on *computer* system architecture, rather than on *organizational* system architecture.)

The Hardware Component

The most visible part of the computer system is obviously the hardware that makes up the system. Consider the computer system upon which you write and execute your programs. You use a keyboard or touch screen and a pointing device to provide **input** of your program text and data, as well as for commands to the computer. A display screen is commonly used to observe **output**. A printer is frequently available as an alternative output to the screen. These are all physical components.

Calculations and other operations in your program are performed by one or more **central processing units (CPUs)**, or “cores” inside the computer. **Memory** is provided to hold your programs and data while processing is taking place. Other I/O devices, such as an SSD, disk, and SD plug-in cards, are used to provide long-term storage of your program and data files. Data and programs are transferred between the various I/O devices and memory for the CPUs to use.

The CPUs, memory, and all the input, output, and storage devices form the **hardware** part of a computer system. The hardware forms the tangible part of the system. It is physical—you can touch it, which is what the word “tangible” means. A typical hardware block diagram for a computer is seen in Figure 1.7. In addition to the I/O devices shown in this diagram, Figure 1.8 lists some other I/O devices that are frequently seen as part of computer systems. The diagram in Figure 1.7 actually applies equally well to large mainframe computers, small personal computers, cell phones, and tablets, and even devices with computers embedded in them, such as automobiles, smart doorbells, and home voice recognition devices. Large and small computers differ primarily in the number of cores, the amount of memory, speed, capacity, and the selection of I/O devices provided. The basic hardware components and design are very similar.

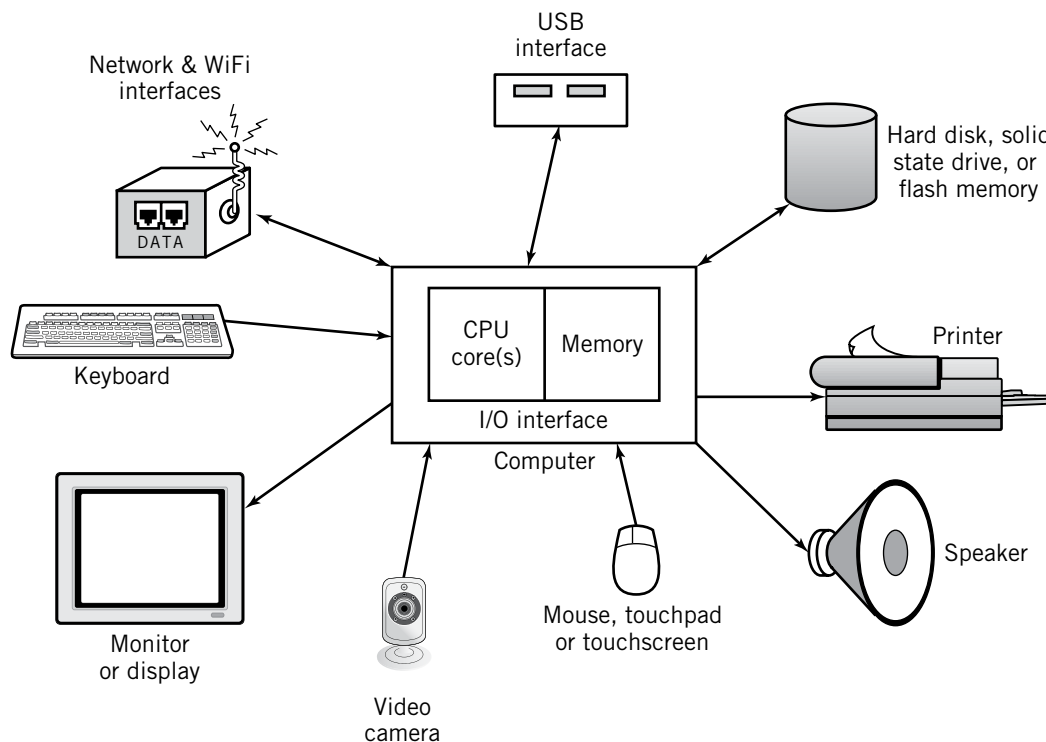


FIGURE 1.7
A Typical Personal
Computer System

- Page and document scanners
- RFID and NearFieldCommunication readers
- TV and radio tuners
- GPS receivers
- Cellular and Bluetooth communication technology
- SD, SmartCard, etc. card readers
- Fingerprint readers
- Graphics tablets
- Other mobile devices: accelerometers, gyroscopes, etc.

FIGURE 1.8 Other Common Input/Output Devices

Conceptually, a CPU itself is often viewed as a composition of three primary subunits:

1. The **arithmetic/logic unit (ALU)** where arithmetic and Boolean logical calculations are performed.
2. The **control unit (CU)**, which controls the processing of instructions and the movement of internal CPU data from one part of the CPU to another.
3. The **interface unit**, which moves program instructions and data between the CPU and other hardware components.

(In modern CPUs, the actual implementation is usually modified somewhat to achieve higher performance, although the basic concepts are carefully preserved. More about that later, in Chapter 8.)

The interface unit interconnects the CPU with memory and also with the various I/O modules. It can also be used to connect multiple CPU cores together. In many computer systems, a bus interconnects the CPU, memory, and all of the I/O components. A **bus** is simply a bundle of wires that carry signals and power between different components. In other systems, the I/O modules are connected to the CPU through one or more separate processors known as **channels**.

The main memory, often known as primary storage, working storage, or **RAM** (for **random access memory**), holds programs and data for access by the CPU. **Primary storage** is made up of a large number of cells, each numbered and individually addressable. Each cell holds a single binary number representing part of a data value or part of an instruction. The smallest addressable size of the cell in most current computers is 8 bits, known as a **byte** of memory. Eight bits of memory can only hold 256 different patterns, so neighboring cells in memory are nearly always combined to form groupings with a larger number of bits. In many systems, for example, 4 bytes of memory combine to form a 32-bit **word**. Modern computers address memory at least 4 bytes (a “32-bit” computer) or 8 bytes (a “64-bit” computer) at a time to take advantage of larger instruction and data groupings.

The amount of primary storage determines the maximum number of instructions and data words that can be loaded into memory from a peripheral device at one time. For example, a computer with 8 gigabytes (GB), actually 8,589,934,592 bytes,² of memory would not be able to execute a program that requires 9.7 GB for its instructions and data unless some means is provided within the computer to load the program in sections as each section of the program is needed.

²1 kilobyte actually equals 1024 bytes. Thus, 1 MB = 1024 × 1024 = 1,048,576 bytes × 8192 = 8,589,934,592 bytes.

The amount of primary storage provided in a typical computer has increased rapidly as computer technology improves. Whereas 64 *kilobytes* (KB) of memory was considered a large amount in 1980, even the least expensive personal computers today usually have 8 *gigabytes* (GB) of memory or more. Large computers may provide many terabytes of primary storage. There are programs on the market that require hundreds of megabytes (MB) or gigabytes (GB) of memory to execute. The inexpensive availability of increased amounts of memory has allowed the design of very sophisticated programs that would not have been possible just a few years ago.

The same is true for secondary storage. Even small personal computers provide long-term storage using hard disks or solid-state storage devices with storage measured in hundreds or thousands of gigabytes. The storage of images and video, in particular, requires tremendous amounts of storage capacity. It is not uncommon to see systems providing tens or hundreds of trillions of bytes (specified as *terabytes*) of long-term storage.

The instructions that form a particular program are stored within the primary storage, then brought into the CPU and executed. Conceptually, instructions are brought in and executed one at a time, although modern systems overlap the execution of instructions to some extent. Instructions must be in primary storage in order to be executed. The control unit interprets each instruction and determines the appropriate course of action.

Each instruction is designed to perform a simple task. Instructions exist to perform basic arithmetic, to move data from one place in the computer to another, to perform I/O, and to accomplish many other tasks. The computer's power comes from the ability to execute these simple instructions at extremely high speeds, measured in billions or trillions of instructions executed per second. As you are already aware, it is necessary to translate high-level language programs into the language of the machine for execution of the program to take place. It may require tens, hundreds, or even thousands of individual machine instructions to form the machine language equivalent of a single high-level language statement. Program instructions are normally executed sequentially, unless an instruction itself tells the computer to change the order of processing. The instruction set used with a particular CPU is part of the design of the CPU and cannot normally be executed on a different type of CPU unless the different CPU was designed to be instruction set compatible. However, as you shall see, most instruction sets perform similar types of operations. As a result, it is possible to write programs that will **emulate** the instruction set from one computer on a computer with a different instruction set, although a program written for the original machine may execute slower on the machine with the emulator.

The data that is manipulated by these instructions is also stored in memory while being processed. The idea that the program instructions and data are both stored in memory while being processed is known as the **stored program concept**. This important concept is attributed primarily to John von Neumann, a famous computer scientist. It forms the basis for the computer architecture that is standard to nearly every existing computer.

The Software Component

In addition to the hardware requirement, your computer system also requires **software**. Software consists of the programs that tell the computer what to do. To do useful work, your system must execute instructions from some program.

There are two major categories of software: system software and application software. System software helps you to manage your files, to load and execute programs, and to accept your commands. The system software programs that manage the computer are collectively known as an **operating system**, and differ from the application programs, such as Microsoft Word or Facebook or Zoom or the programs that you write, that you normally

run to get your work done. Windows, Linux, macOS, iOS, and Android are the best known current examples of an operating system. Others include UNIX and IBM z/OS.

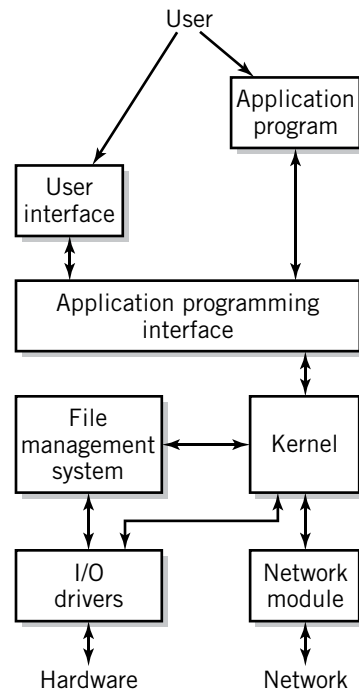
The operating system is an essential part of the computer system. Like the hardware, it is made up of many components. A simplified representation of an operating system is shown in Figure 1.9. The most obvious element is the user interface that allows you to execute programs, enter commands, and manipulate files. The user interface accepts input from a keyboard and, in most modern systems, a mouse, touch screen, or other pointing device. The user interface also does output presentation on the display. On some systems, the output display might be simple text, but more likely the display includes a graphical user interface with a windowing system, and various gadgets for manipulating the windows.

The operating system's **application programming interface (API)** acts as an interface for application programs and utilities to access the internal services provided by the operating system. These include file services, I/O services, data communication services, user interface services, program execution services, and more.³

Many of the internal services are provided by the **kernel** module, which contains the most important operating system processing functions. The remaining services are provided by other modules that are controlled by the kernel. The kernel manages memory by locating and allocating space to programs that need it, schedules time for each application to execute, provides communication between programs that are being executed, manages and arranges services and resources that are provided by other modules, and provides security.

The file management system allocates and manages secondary storage space and translates file requests from their name-based form into specific I/O requests. The actual storage

FIGURE 1.9
Simplified
Operating System
Block Diagram



³The same term (API) is also used to describe the services provided by one application to another. For example, Amazon and Google are among many companies whose application software provides API tools to allow users to extend the functionality of the original software.

and retrieval of the files is performed by the I/O drivers that comprise the I/O component of the operating system. Each I/O driver controls one or more hardware devices of similar type.

The network module controls interactions between the computer system and the network(s) to which it is attached.

Traditionally, the operating system software has nearly always been stored on a hard disk, but on some smaller modern systems, especially lightweight laptops and embedded systems such as cell phones, tablets, and e-books, a solid-state drive or SD card is normally used instead. On a few systems the operating system is actually provided as a network or cloud-based service when the system is turned on. In either case, the bootstrap or IPL (Initial Program Load) program in the operating system is stored within the computer using a type of memory known as **ROM**, or **read-only memory**. The bootstrap program provides the tools to test the system and to load the remainder of the operating system from the disk or network. Although the physical medium where the software is stored can be touched, the software itself is considered intangible.

Together, the hardware and system software provide a working computer system environment. Application software, communication support, and user data complete the picture.

The Communication Component

Very few modern computers or computer-based devices operate independently. Instead, they are usually designed to interact with other computers. To work together, computers must have means to communicate with each other through a communication connection of some sort. The computers may be located physically close to each other, or they may be separated, even by thousands of miles. The communication component requires both hardware and software to achieve this goal. Additional hardware components physically connect computers together into multiprocessing systems, or clusters, or networks, or, via telephone, satellite, microwave, Internet, or other network technology, to computers at other remote locations. A **communication channel** provides the connection between computers. The channel may be a wire cable, a fiber-optic cable, a telephone line, a wireless technology, or some combination of these. Examples of wireless technologies include infrared light, cellular phone, satellite, Wi-Fi, and Bluetooth. Special I/O hardware, consisting of a device such as a cellular **modem** or **network interface card (NIC)** within the computer, serves as an interface between the computer and the communication channel. There may be additional hardware within the channel itself.

The communication component also requires additional software within the operating system of each computer to make it possible for each computer to understand what the other computers that they are connected with are saying. This software establishes the connections, controls the flow of data, and directs the data to the proper applications for use.

The Computer System

To review, our general description of the computer is valid for all general-purpose computer systems, and also for most devices with computers embedded in them, regardless of brand name, usage, or size. In more general terms, every computer system consists of at least one CPU, where all the processing takes place; memory to hold the programs and data while they are being processed; and some form of I/O, usually one or more keyboards, touch screens, pointer devices, and flat-screen display devices. Computer systems also include one or more forms of long-term storage, usually hard disks or solid-state storage, but also

possibly network (“cloud”) storage, optical disks, and USB or SD plug-in memory. Most modern computer systems provide more than one CPU (or “core”) within the computer system. A single CPU can process only one instruction at a time; the use of multiple CPUs can increase processing speed by allowing instructions that do not affect each other to be executed in parallel.

The validity of our general description is true regardless of how complex or simple the computer system may seem.

As a specific example, the large z15 T01 IBM mainframe computer shown in Figure 1.10 can provide complex Web services to thousands of users at a time. IBM z15 mainframes can have as many as 190 CPUs working together, with a minimum of 512 GB up to 40 terabytes (TB) of primary storage. They are capable of executing instructions at rates of up to hundreds of billions of instructions per second! The powerful z/OS operating system can keep track of hundreds or thousands of simultaneous users and divides the time among them to satisfy their differing requirements. In addition to the CPU, there are many large I/O devices—including tape drives and high-speed printers—and disks and solid-state devices that store essentially unlimited amounts of data. The biggest version of this computer alone weighs over 6000 pounds/2800 kilograms!

In contrast, the Apple iPad tablet shown in Figure 1.11 is designed for personal use. Everything is self-contained in one hand-held package. This system only has 3 GB of primary RAM storage and operates at a small fraction of the speed of the z15. Long-term storage is limited to 128 GB of solid-state memory. The entire system, complete with display screen, built-in webcams, multiple network connections, including an optional cellular connection, and battery, weighs about 1.1 pounds (0.49 kilograms, if you prefer).

Although these two systems seem very different, the difference is actually one of magnitude and application, not of concept. The large system operates much faster, can support much more memory, and handles more input and output much faster. It has operating system software that allows many users to share this larger resource. Nonetheless, the fundamental system architecture is remarkably similar in both cases. Even the actual processing performed by the CPU is similar.

In fact, today’s CPU operates in the same fundamental way as its CPU counterpart of sixty years ago, even though the construction is very different. Since computers all operate so similarly, regardless of size or type, it is not difficult today to transfer data between these different systems, allowing each system to do part of the processing for higher overall efficiency. This concept is known as **distributed computing**. The fact that different types of computers can work together, share files, and communicate successfully is known as **open computing**. Communication technology fulfills the requirements that make open and distributed computing possible and convenient.

FIGURE 1.10 IBM System z15 T01 Mainframe Computer





FIGURE 1.11
Apple iPad
Tablet Computer

Computers are sometimes divided into categories: mainframe computers, midsized servers, workstations, personal desktop and laptop computers, and mobile computing devices, but these categories are less significant than they once were. The capability of today's smart-phone far exceeds the capabilities of a mainframe computer of years ago. Indeed, it is not uncommon to see workstations that are being used as though they were midsized servers, or even small mainframes. Rather than attempting to categorize a particular computer, it is usually more productive to describe its capabilities in comparison to other systems being discussed or considered.

1.3 The Concept of Virtualization

The word “**virtual**” appears frequently throughout the computer literature in many different contexts. To name a few applications of the word that appear in this text, there are *virtual* computers, a Java *Virtual Machine* (JVM), *virtual* memory, and *virtual* networks. Sometimes, a synonymous word, **logical**, is used instead: in networking we have *logical* connections. *Virtual* storage consists of a relationship between *logical* memory and physical memory.

It is not important at this point that you understand any of the specific concepts mentioned above. (In fact, we realize that you probably don't.) Since the words *virtual* and *logical* represent a number of important concepts in IT, however, we introduce them here.

In optics, a virtual image is the reflection that you see when you stand in front of a regular mirror. (See, for example, the cartoon at the beginning of Chapter 19.) You know that the image isn't real. For one thing, it's behind the wall that the mirror is mounted on. For another, you can't touch it. Another example that might be familiar to you is *virtual reality*, the technology that allows people to see and interact with things that aren't there. In early, time-shared computing, a large central computer commonly supplied computing services to users at terminals located remotely from the computer. In a sense, it seemed as though the user had access to a computer that was all her own. Starting in the early 1970s, IBM offered the VM (*virtual machine*) operating system to support this concept. (The centralized time-sharing approach is similar, in some ways, to today's cloud computing—one of the goals of this text is to convince you that many of today's “new and exciting” technologies are simply reworkings of ideas that have been around for a long time!)

The *American Heritage Dictionary* offers two applicable definitions of *virtual* that together describe the usage of the word in modern computing:

- existing or resulting in essence or effect though not in actual fact, form, or name;
- created, simulated, or carried on by means of a computer or computer network.

In essence, *virtual* and *logical* are used to refer to something that appears as though it is something different. Thus, the *Java Virtual Machine* uses software to simulate a real computer that works well with the Java programming language, even though the actual computer executes a different set of instructions than the JVM. A *logical connection* in networking offers the appearance of a direct communication link for passing data between two computers, even though the actual connection might involve a complex series of interconnections involving many computers and other devices and a variety of software to make it all look simple. The *virtualization* of a computer allows a single computer to appear as a multiplicity of computers, each with its own operating system and hardware resources. A single mainframe set up as a cloud service might provide hundreds or thousands of virtual computers to users all over the world.

1.4 Protocols and Standards

Standards and protocols are of great importance in computer systems. **Standards** are agreements among interested parties, often manufacturers, to assure that various system components will work together interchangeably. Standards make it possible to build a computer with components from different suppliers, for example, knowing that a graphics card will plug properly into a connector on a motherboard and that the image representations will be consistent between the connector, the CPU, memory, and the display monitor.

Standards apply to every aspect of computing: hardware, software, data, and communications; the voltage of a power supply; the physical spacing of pins on a connector; the format of a file; and the pulses generated by a mouse. Computer language standards, such as Java and SQL, allow programs written on one type of computer to execute properly and consistently on another, and also make it possible for programmers to work together to create and maintain programs.

Similarly, data format and data presentation standards, such as the PNG and JPEG image format standards, the Unicode text format standard, and the HTML and XML Web presentation standards, allow different systems to manipulate and display data in a consistent manner.

Standards can arise in many different ways. Many standards occur naturally: a proprietary data format, belonging to a single vendor, becomes a *de facto* standard due to the popularity of the product. The PDF print description language is an example of such a standard. The format was designed by Adobe Corporation to provide a way of communicating high-quality printed output between computers and printers. Other standards are created because of a perceived need in an area where no standard exists.

Often a committee will form to investigate the requirements and create the standard. The MPEG-2 and MPEG-4 standards, which establish the means for the transmission and processing of digital video images, occurred in this way. The committee that designed the standard, made up primarily of motion picture engineers and video researchers, continues to develop the standard as improved techniques evolve. The JPEG photographic standard and MP3 and MP4 sound standards are other examples of standards that were developed formally. Similarly, each version of HTTP has been formalized after many years of discussion by parties interested in Web communication. A nonstandard protocol or data format is limited in use to its supporters and may or may not become a standard, depending on its general acceptance. For example, DVD videos encoded in the proprietary DVD-ROM format will play on some DVD players, but not on others.

Protocols define the specific agreed-upon sets of ground rules that make it possible for a communication to take place. Except for special applications, most computers perform their operations such that each hardware or software computer unit will understand what other